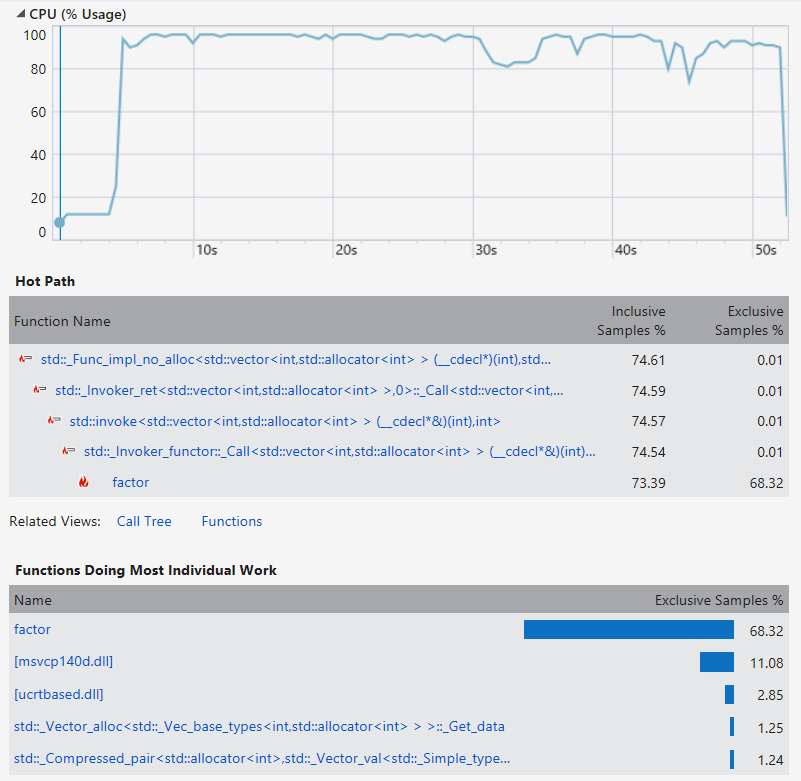
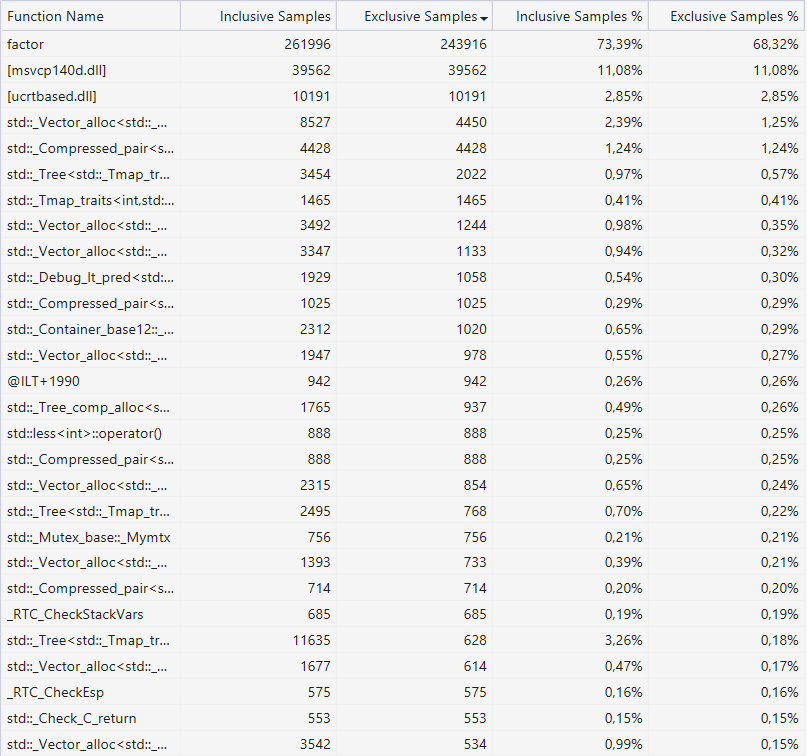
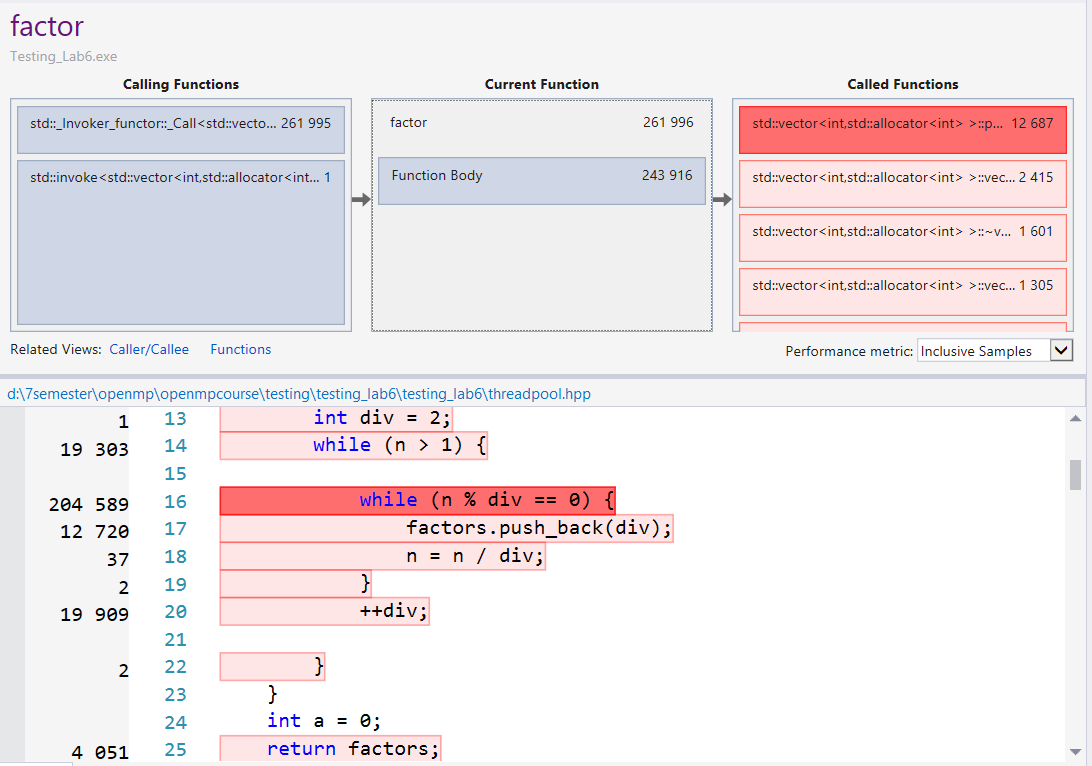
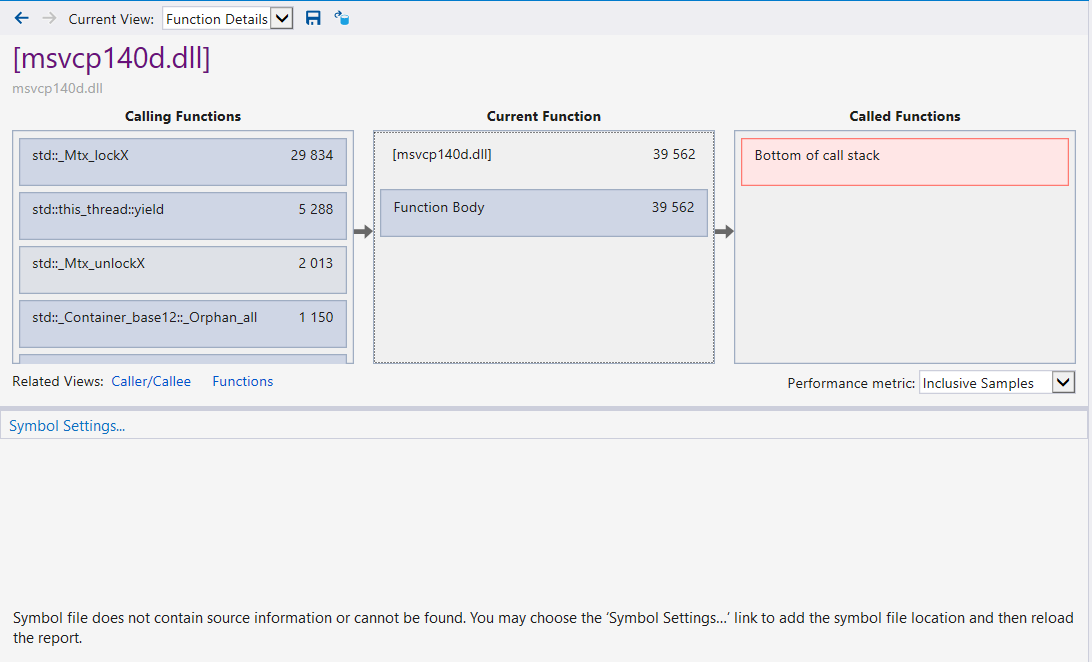
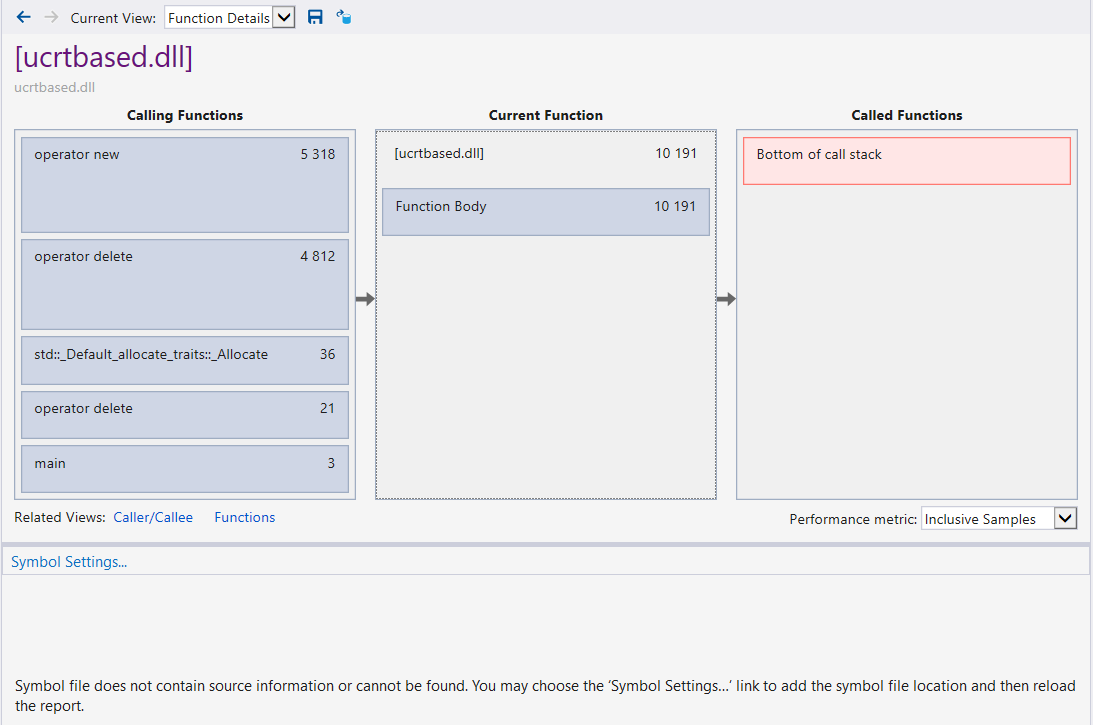
1. Провести профилирование с помощью **дискретного профилировщика**. В отчет занести результаты из представлений Summary, Functions. Для функций, на выполнение которых затрачено больше всего процессорного времени, добавить в отчет информацию из представления Function Details. Для данных в каждом из этих представлений написать комментарии.

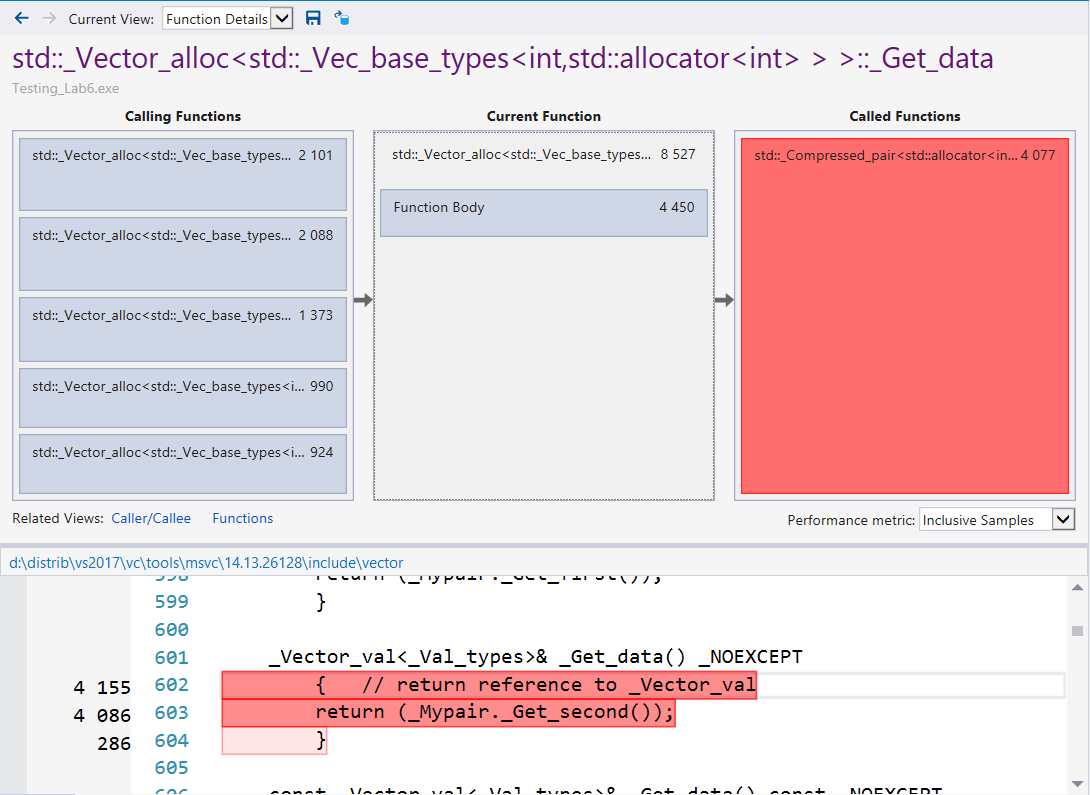






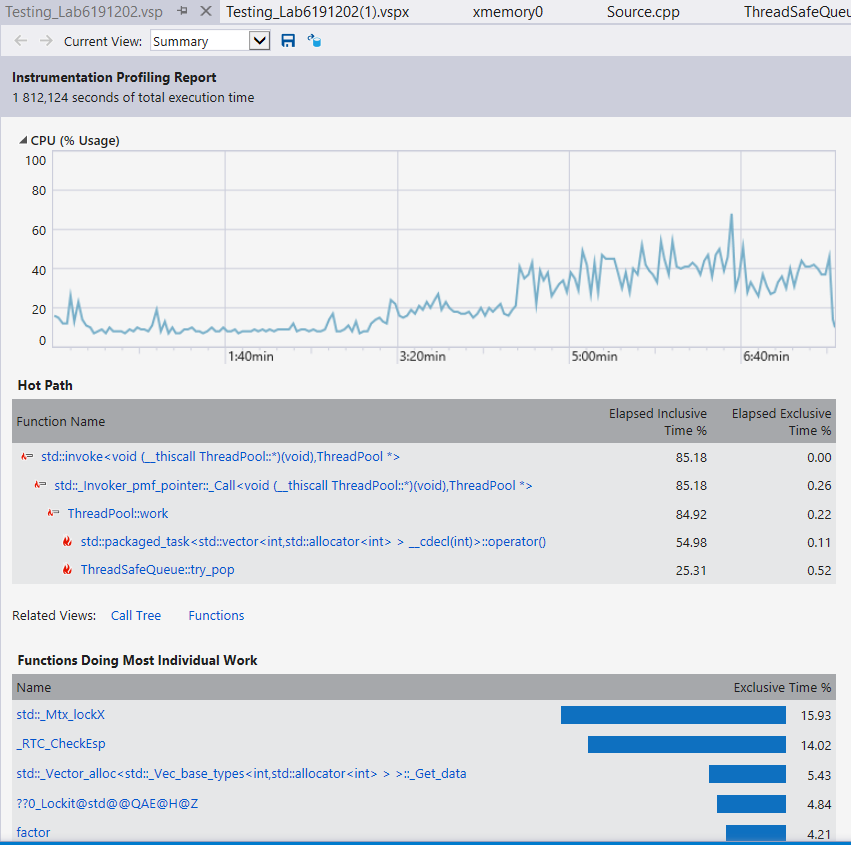


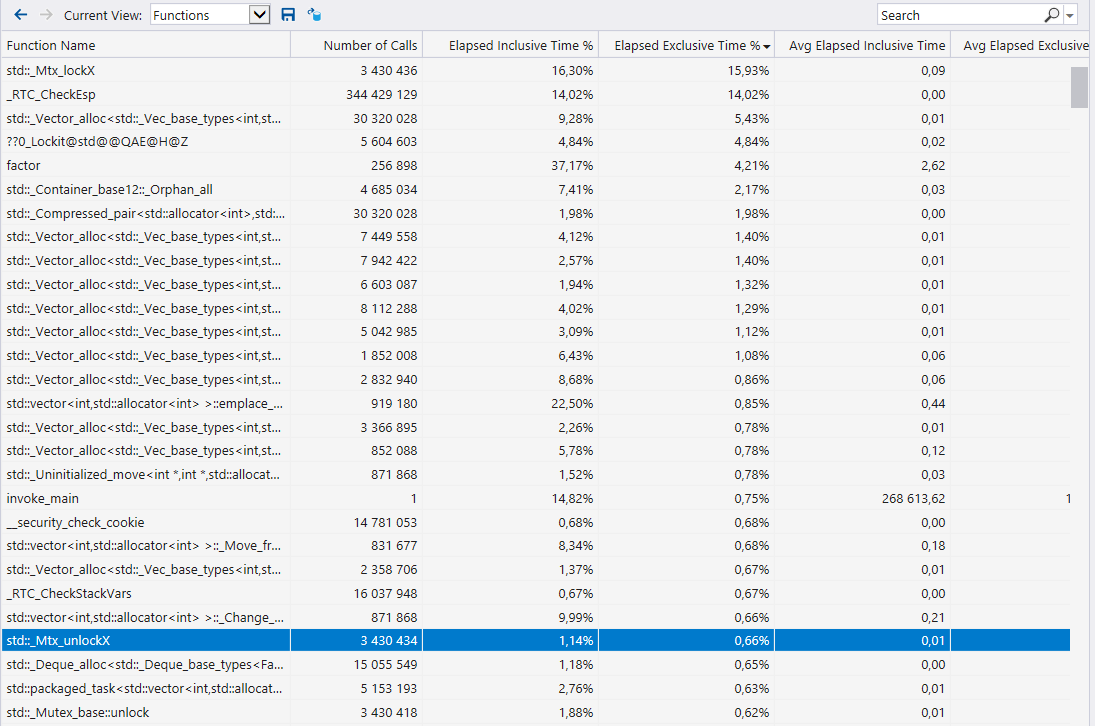




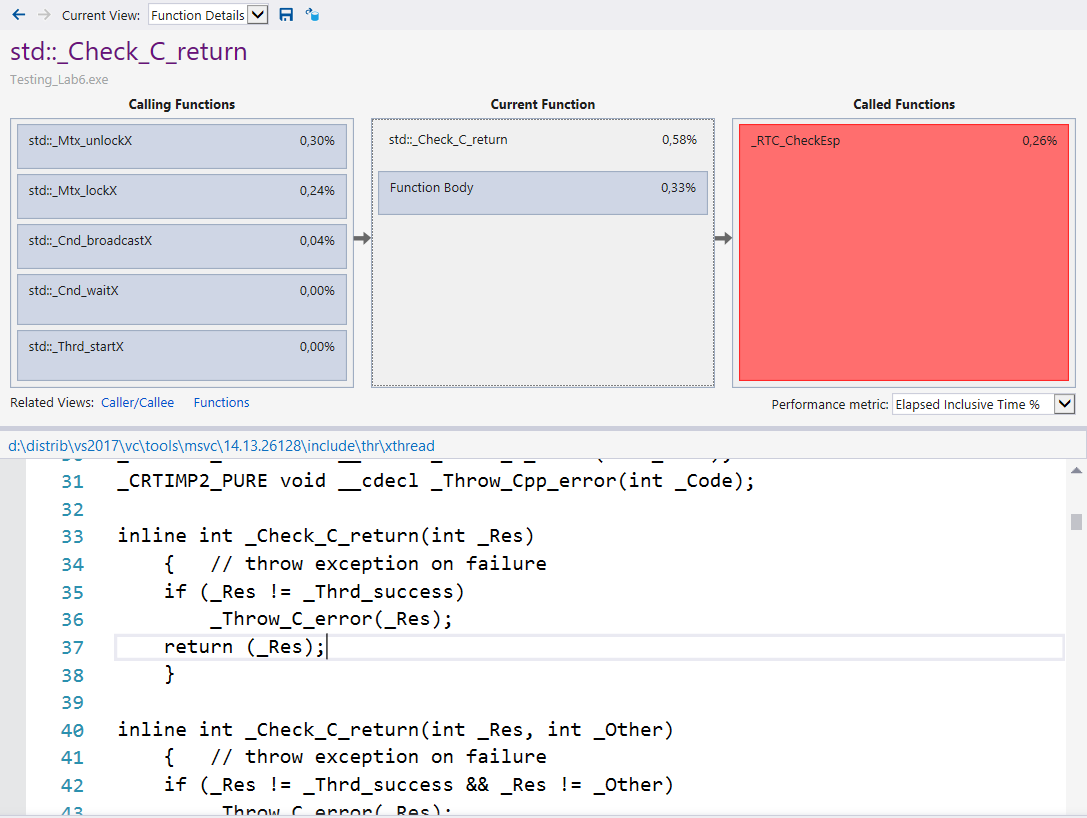
Наибольшее число попаданий приходится на функцию factor, которая осуществляет разложение аргумента на простые множители, а также на функции из заголовочного файла vector. Это скорее всего связано с тем, что простые множители заносятся в vector, который затем возвращается в future.

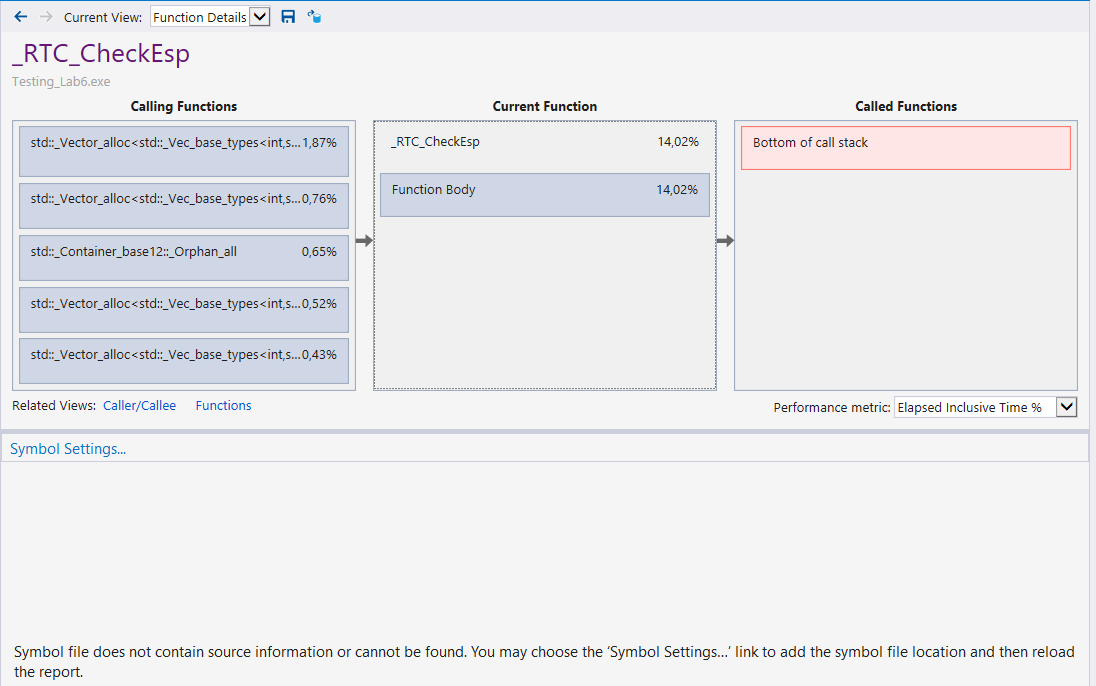
1. Провести профилирование с помощью **инструментированного профилировщика**. В отчет занести результаты из представлений Summary, Functions. Для функций, у которых наибольшее исключительное время, добавить в отчет информацию из представления Function Details. Для данных в каждом из этих представлений написать комментарии.

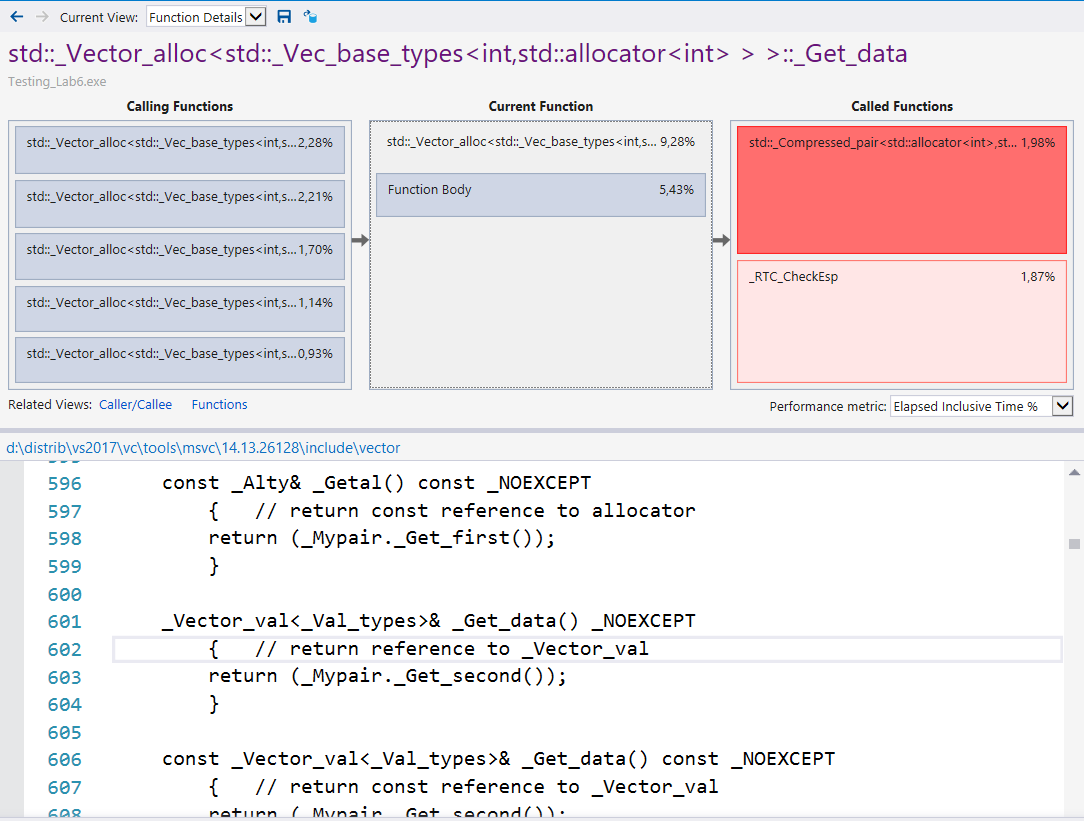




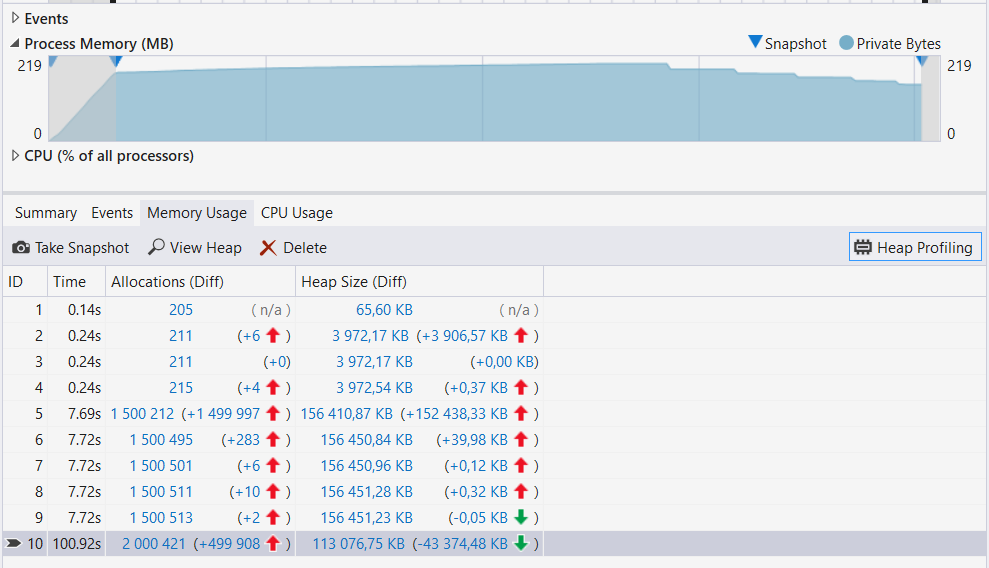
Процент попаданий для методов заметно отличается от результатов дискретного профилировщика. Большое число попаданий приходится на функцию, которая судя по всему связана с захватом мьютекса, который используется в очереди задач для безопасного извлечения элементов. Также на функции, связанные с вектором и функцию factor.

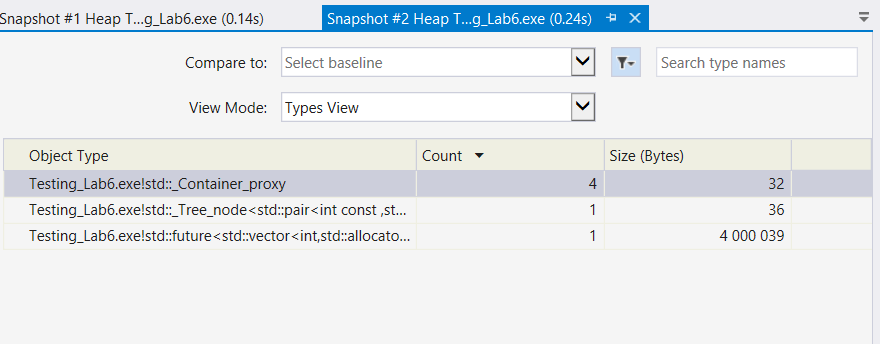




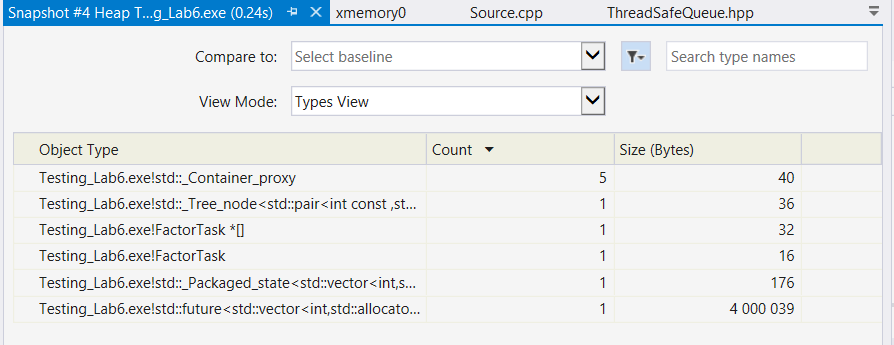


1. Провести профилирование с помощью **профилировщика выделения памяти**. В отчет занести результаты из представлений Summary, Functions. Для функций, которые выделили больше всего памяти (исключительное значение), добавить в отчет информацию из представления Function Details. Также в отчет включить информацию из представления Allocation, а именно какие ветви в стеке вызовов выделили памяти больше всего. Для данных в каждом из этих представлений написать комментарии.

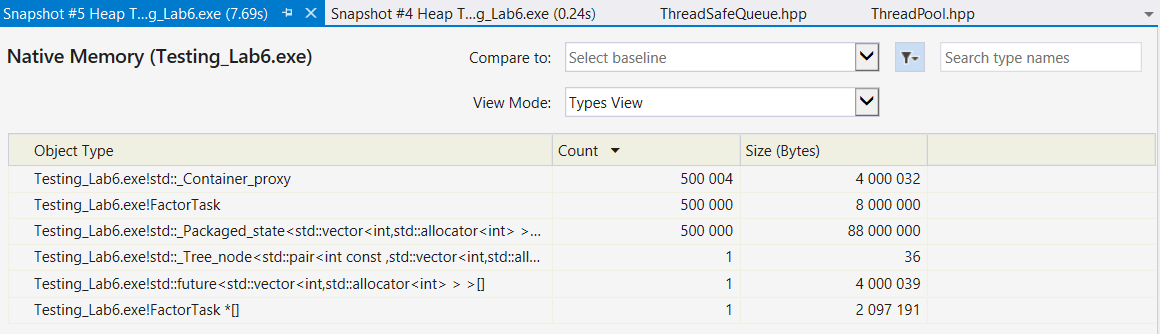




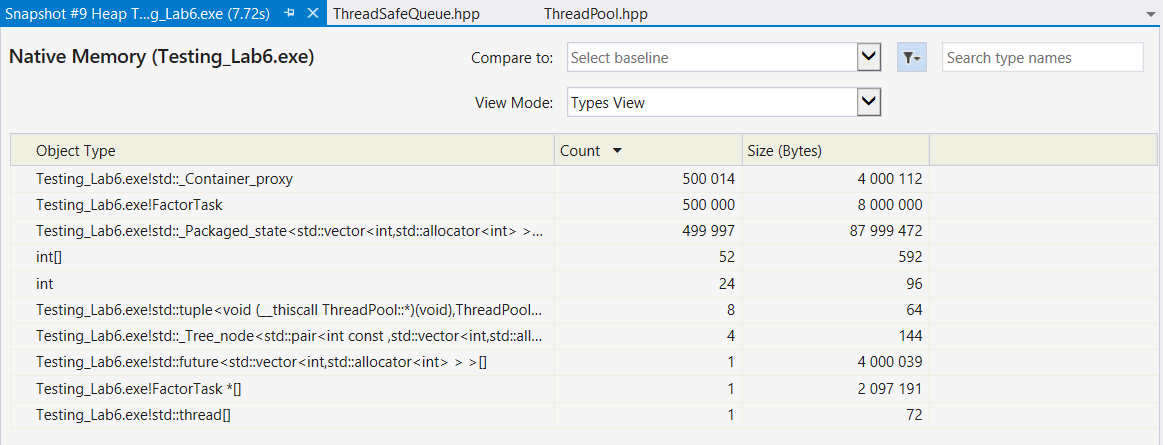
В куче зарезервирована память для вектора futures (500 000).



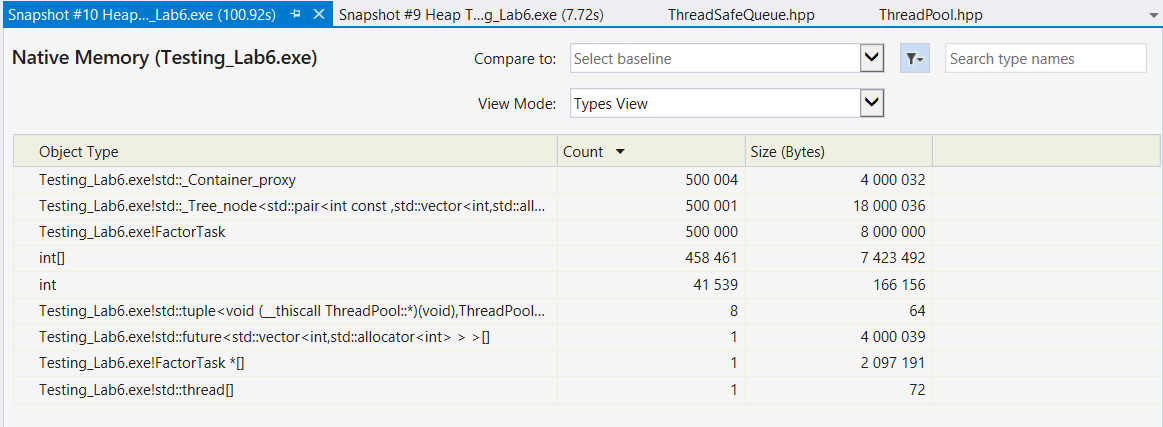
Задействуется функция submit. Выделена память для структуры FactorTask, а также для объекта packaged\_task.



Цикл с созданием тасков завершен. Выделена память для 500 000 элементов.

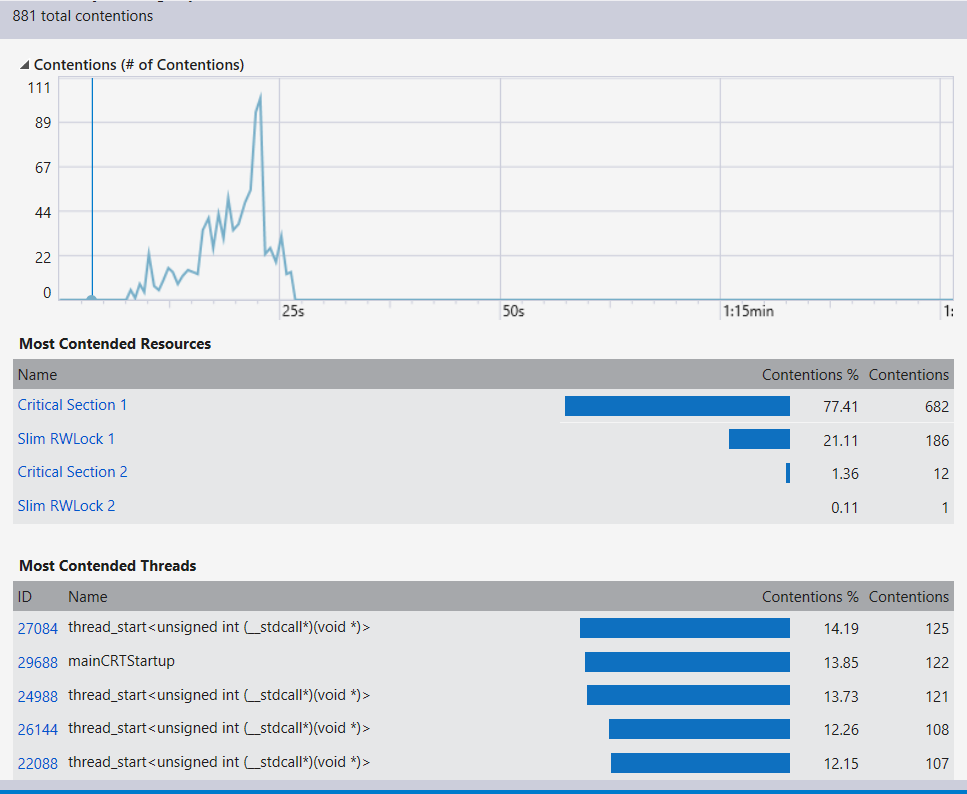


Снэпшот одного из шагов цикла внутри метода start() пула. Потоки, которые были к этому моменту запущены, уже начали выполнение тасков и соответственно была выделена память для массивов типа int (которые содержатся внутри структуры vector ), элементы которых – простые множители. А также выделилась память для просто чисел типа int (видимо это для случая, когда vector содержит один элемент).



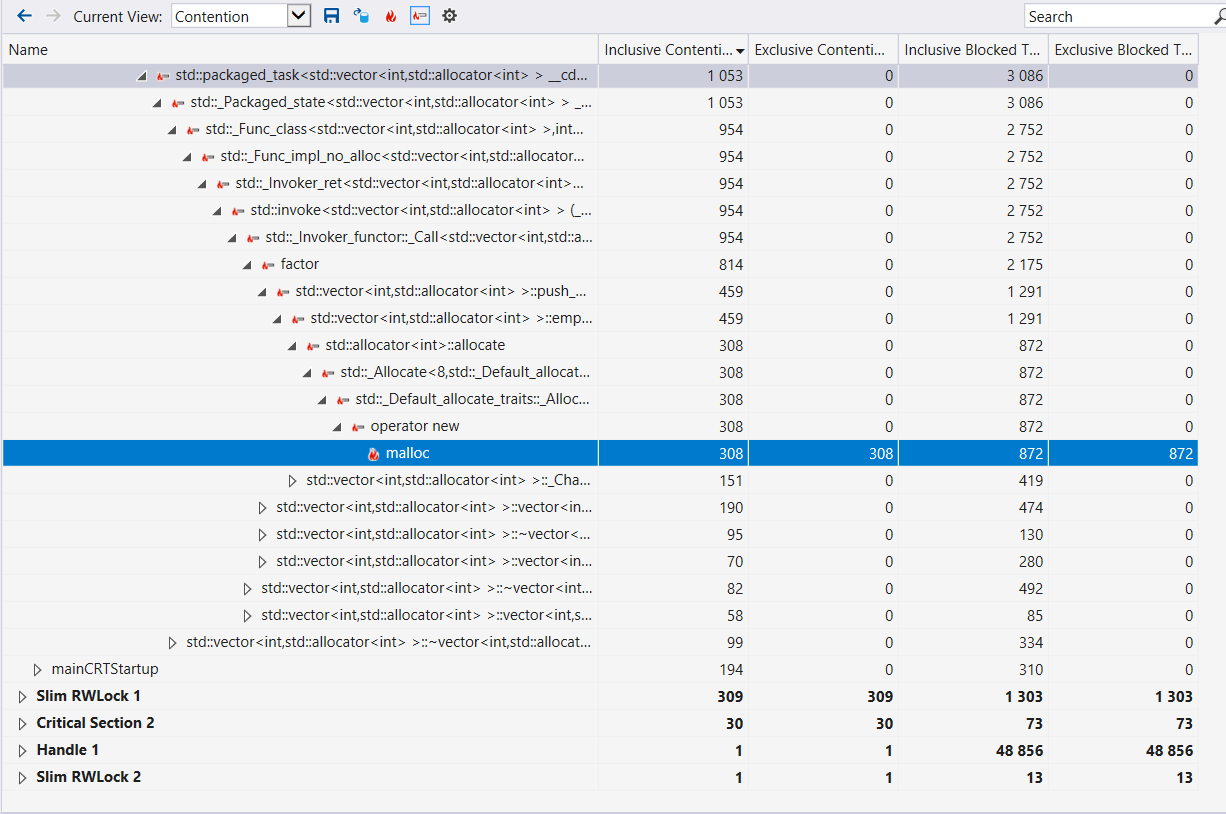
В сумме всего получается 500 000 элементов типа int или int[].

1. Провести профилирование с помощью **профилировщика конкуренции**. Для режима Concurrency в отчет занести результаты из представления Resource Details, Threads Details. Для режима Concurrency Visualizer – сводную информацию. Добавить комментарии.

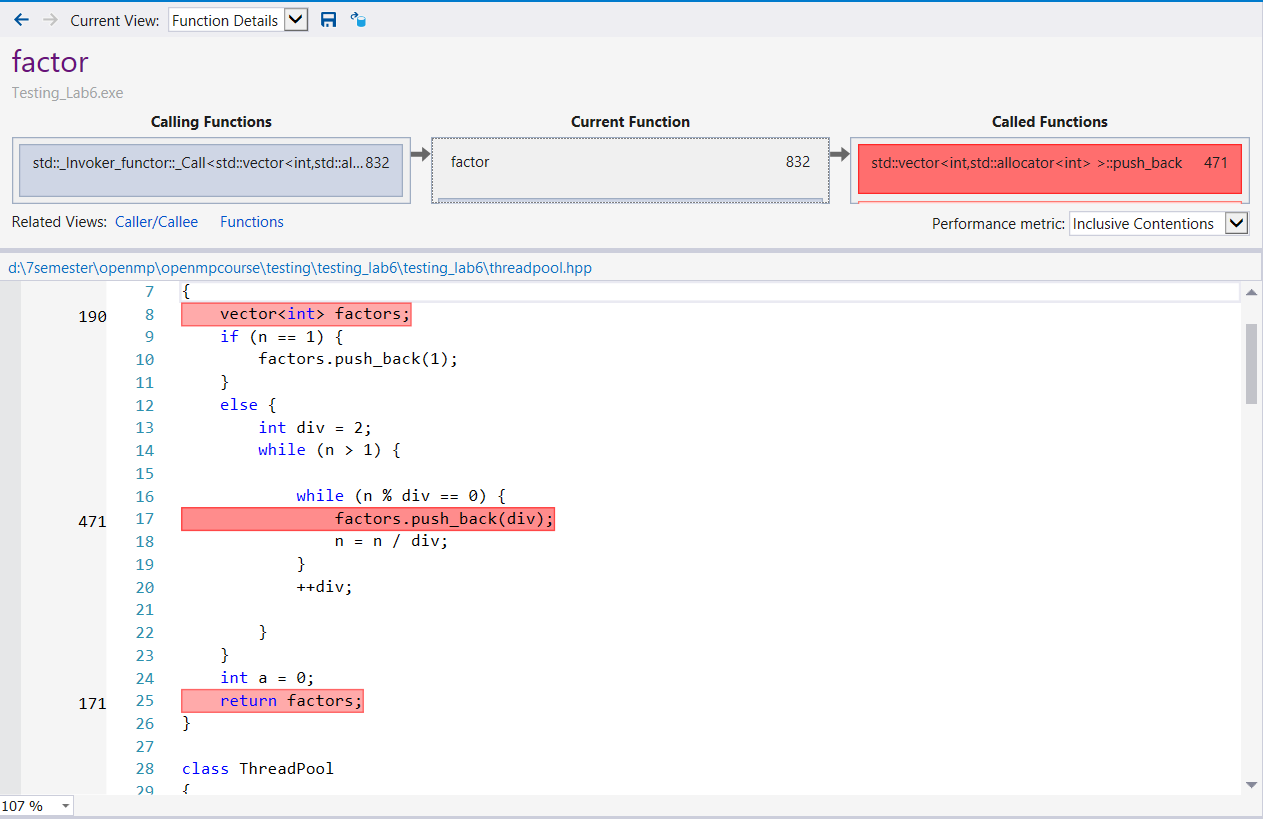


По диаграмме можно увидеть, что пики конкуренции приходятся на первую половину времени выполнения. Наибольшая конкуренция происходит за ресурсы Critical Section 1 и Slim RWLock 1.

В представлении Contention для ресурса можно увидеть, с чем он связан.

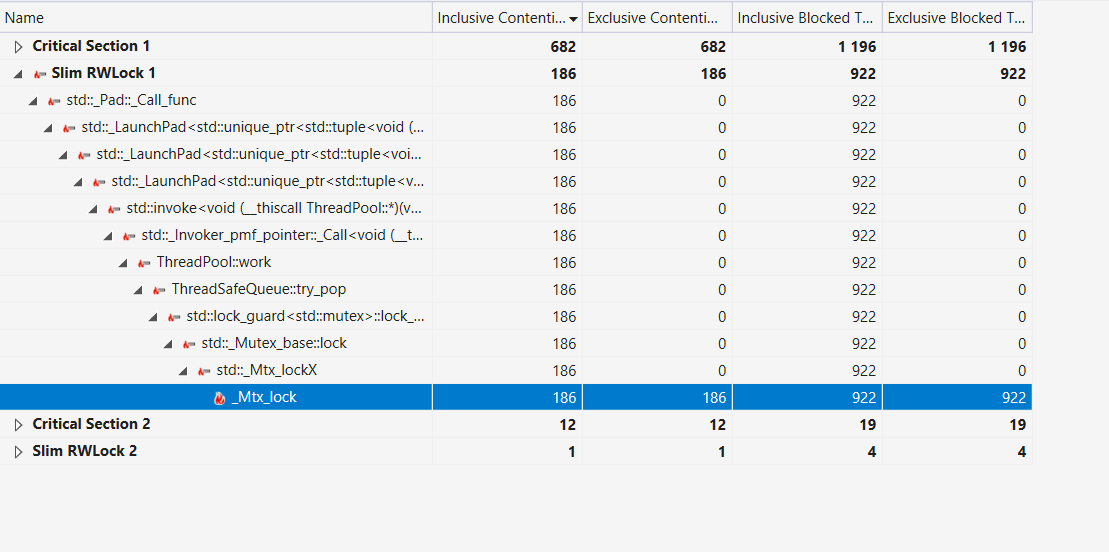


Судя по всему, critical section 1 связана с выделением памяти для вектора

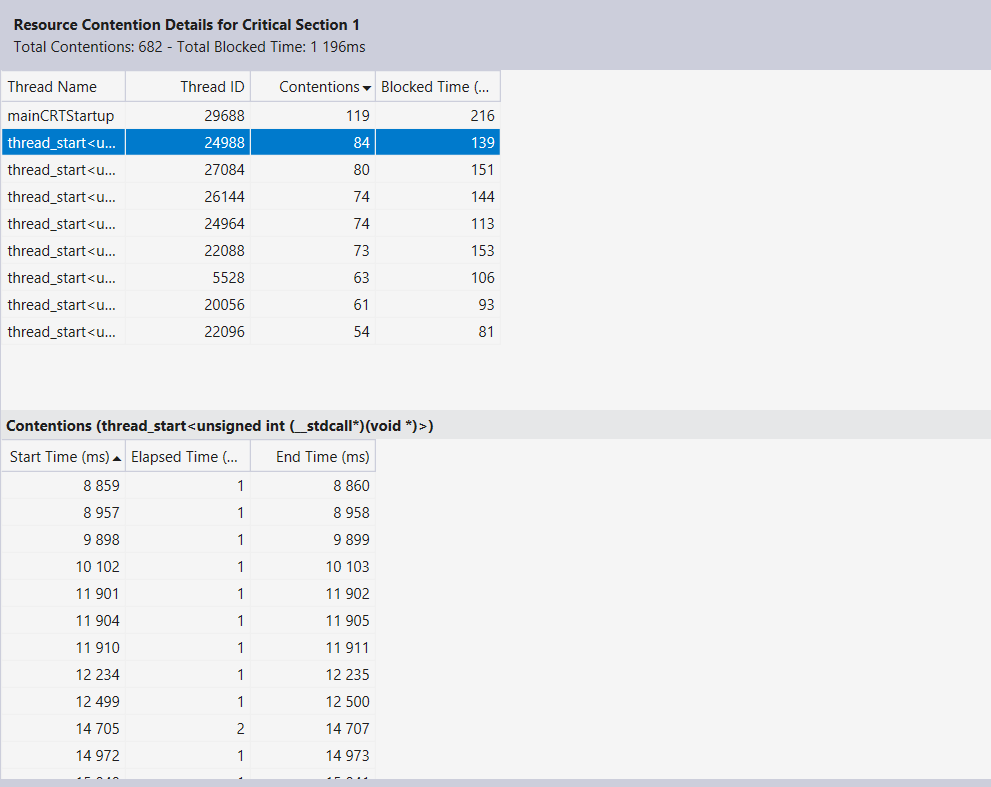


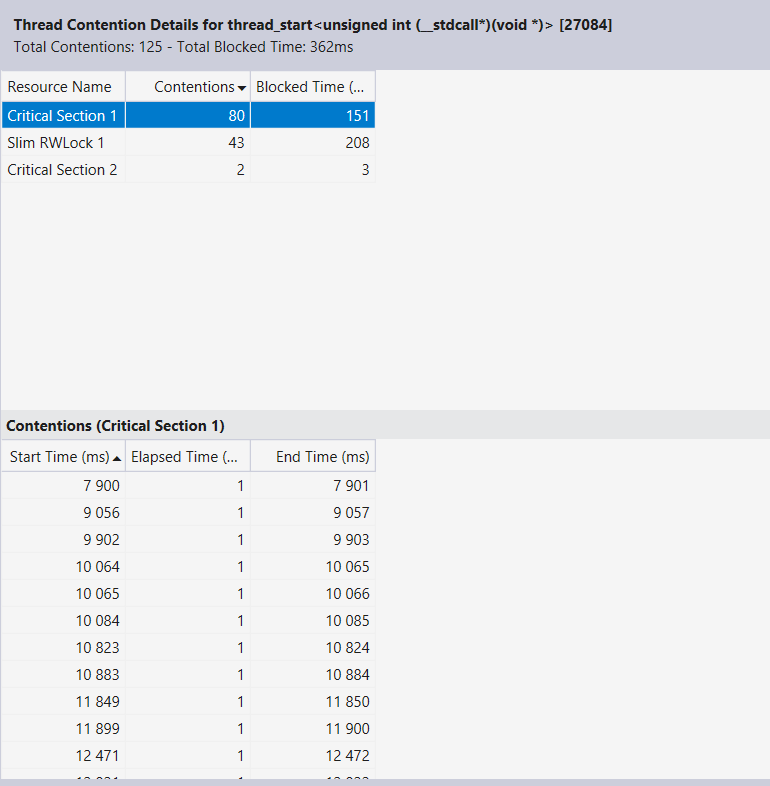
Наибольшая конкуренция происходит при одновременном выделении памяти потоками для структуры vector.

Представление Contention для ресурса Slim RWLock 1

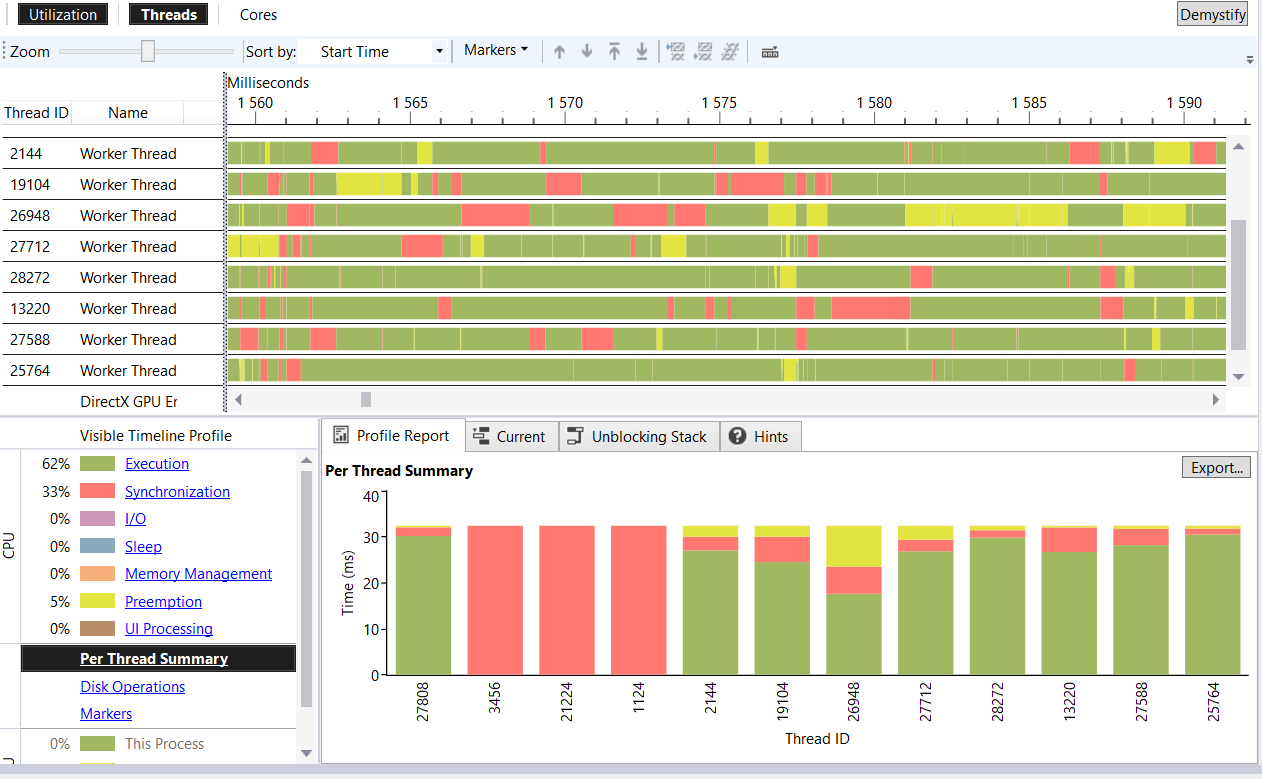


Этот ресурс связан с потокобезопасной очередью ThreadSafeQueue. При попытке одним потоком извлечения элемента из очереди происходит захват мьютекса, чтобы другие потоки не имели на этот момент доступа к очереди.





Concurrency Visualiser



На диаграмме можно увидеть, что в первой половине времени выполнения рабочие потоки большую часть времени находятся в состоянии выполнения, и иногда в состоянии синхронизации (когда ожидают освобождения ресурса), либо в состоянии preemption. Но большую часть времени они выполняют полезную работу.

